REMARKS

Claims 1-44, 46, 47, 50, and 51 have been cancelled to be prosecuted in a subsequent continuing application(s), in affirmation of applicant's election of the Examiner's Group II Claims. Moreover, Claims 45, 48, 49, 52, and 53 have been amended to clarify what is claimed. Still yet, Claims 54-62 have been added to include subject matter which is supported by the originally filed specification and which applicant believes to be patentable. Claims 45, 48, 49, 52, 53, and 54-62 are currently pending. In the event a telephone conversation would expedite the prosecution of this application, the Examiner may reach the undersigned at (408) 505-5100.

Claims 45 and 52 are rejected under 35 U.S.C. 102(e) as being anticipated by Deering et al., U.S. Patent Number 6,424,343 B1. Such rejection is deemed overcome by way of the amendments made hereinabove. In particular, subject matter similar to that currently pending in Claims 48, 49, and 53 have been incorporated into Claims 45 and 52. Claims 45 and 52 are thus deemed allowable for reasons set forth hereinbelow.

Claims 48, 49 and 53 are rejected under 35 U.S.C. 103(a) as being unpatentable over Deering in view of Struble, "Assembler Language Programming; The IBM System/370 Family," 1984, pages 68-89. Applicant respectfully disagrees with such rejection, especially in view of the amendments made hereinabove.

In particular, only applicant teaches and claims "performing programmable operations on the graphics data utilizing the hardware graphics accelerator in order to generate output, wherein the operations are programmable by a user utilizing instructions from a predetermined instruction set capable of being carried out by the hardware graphics accelerator... [and] the operations include a set on less operation," or similar language. See Claims 45, 48, 49, 52, and 53.

The Examiner admits that Deering fails to disclose an instruction set including load, move, multiply, an addition and a set less than. While applicant agrees with such interpretation, the Examiner continues by stating that Struble discloses in assembler language, programming for

graphics data instructions to perform operations such as load, move, add, subtract and multiply. Moreover, the Examiner states that it would have been obvious to one of ordinary skill in the art at the time of the invention to include in the instruction set of Deering load, move, add, subtract and multiply operations, for basic storage manipulation and movement of graphics data from one memory to another.

Applicant respectfully disagrees with such assertions. In particular, applicant emphasizes that, while Deering teaches a graphics system, Struble discloses a general-purpose processor assembler language. "In order to rely on a reference as a basis for rejection of an applicant's invention, the reference must either be in the field of applicant's endeavor or, if not, then be reasonably pertinent to the particular problem with which the inventor was concerned." See MPEP 2141.01(a). To simply glean features from the art of general-purpose processor assembler languages and combine the same with the non-analogous art of graphics systems would be improper and frustrate the inventive concepts of applicant, especially in view of the fundamentally different problems which the two arts address.

Moreover, the Examiner states that "Struble discloses in assembler language, programming for graphics data...." This is simply untrue. Struble makes simply no reference to graphics data, graphics processing, etc. and, instead, discloses an archaic general purpose-processor for general purpose processing.

More importantly, the Examiner's proposed combination simply fails to meet the specifics of applicant's claims. In particular, neither Deering nor Struble disclose, teach or suggest "a set less than" operation/instruction, as claimed by applicant. Moreover, only applicant teaches and claims "performing programmable operations on the graphics data utilizing the hardware graphics accelerator," "wherein the operations are programmable by a user utilizing instructions from a predetermined instruction set capable of being executed by the hardware graphics accelerator," and "the operations include[ing] a set on less operation (emphasis added)."

Thus, it is clear that applicant's programmable operations/instructions (including the unique "set less than" operation/instruction) is claimed to be "carried out by the hardware graphics

accelerator," a feature found nowhere in the prior art. Applicant further emphasizes that use of the claimed "set less than" operation/instruction is particularly advantageous in the specific art of hardware graphics accelerators, since it facilitates the execution of difficult single-instruction, multiple-data (SIMD) branching in graphics environments. Simply nowhere in the prior art is there taught, disclosed, or suggested such a combination of features for fulfilling the foregoing objectives.

With respect to Claims 49 and 53, the Examiner has rejected the same based upon similar rational as independent Claim 48. Applicant emphasizes, however, that the Examiner has failed to show in the prior art applicant's claimed "transforming and the lighting further include[ing] negating the graphics data and branching." Such features provide advantageous flexibility in the specific context of graphics processing using a hardware graphics accelerator. A showing in the prior art or a notice of allowability is respectfully requested.

In view of the above remarks, all of the independent claims (Claims 45, 48, 49, 52, and 53) are deemed allowable. By virtue of the dependence of the remaining claims on the foregoing independent claims, such dependent claims are also deemed allowable.

The Commissioner is authorized to charge any additional fees or credit any overpayment to Deposit Account No. 50-1351 (Order No. NVIDP021).

Respectfully submitted,

Kevin J. Zilka
Registration No. 41,429

P.O. Box 721120
San Jose, CA 95172-1120
408-505-5100

APPENDIX A

45. (Amended) A method for programmable processing in a [computer graphics pipeline] hardware graphics accelerator, comprising:

[(a)]receiving graphics data including lighting information in a hardware graphics accelerator; and

[(b)]performing programmable operations on the graphics data utilizing the hardware graphics accelerator in order to generate output, wherein the operations are programmable by a user utilizing instructions from a predetermined instruction set capable of being executed by the hardware graphics accelerator;

[(c)]wherein the operations include [a mathematical operation for altering the lighting information of the graphics data] a set on less operation.

48. (Amended) A method for processing graphics data, comprising:

transforming the graphics data utilizing a hardware graphics accelerator; and

lighting the graphics data utilizing the hardware graphics accelerator;

wherein at least one of the transforming and the lighting includes performing operations on the graphics data utilizing instructions from an instruction set capable of being executed by the hardware graphics accelerator, the operations including a no operation, a load, a move, a multiply, an addition, and a set on less than each capable of being carried out by the hardware graphics accelerator.

49. (Amended) A method for processing graphics data, comprising:

transforming the graphics data utilizing a hardware graphics accelerator; and

lighting the graphics data utilizing the hardware graphics accelerator;

wherein at least one of the transforming and the lighting includes performing operations on the graphics data utilizing instructions from an instruction set capable of being executed by the hardware graphics accelerator, the instruction set including a no operation instruction, a load instruction, a move instruction, a multiply instruction, an addition instruction, and a set on less than instruction.

52. (Amended) A method for processing graphics data utilizing a hardware graphics accelerator, comprising:

transforming the graphics data utilizing the hardware graphics accelerator; and

lighting the graphics data utilizing the hardware graphics accelerator;

wherein the transforming and the lighting include performing operations on the graphics data utilizing instructions from an instruction set capable of being executed by the hardware graphics accelerator, the instruction set including a no operation instruction, a load instruction, a move instruction, a multiply instruction, an addition instruction, and a set on less than instruction;

wherein the transforming and the lighting further include negating the graphics data and branching.

53. (Amended) A method for processing graphics data utilizing a hardware graphics accelerator, comprising:

transforming the graphics data utilizing the hardware graphics accelerator, the graphics data including constants; and

lighting the graphics data utilizing the hardware graphics accelerator;

wherein the transforming and the lighting include performing operations on the graphics data utilizing instructions from an instruction set capable of being executed by the hardware graphics accelerator, the instruction set including a no operation instruction, a load instruction, a move instruction, a multiply instruction, an addition instruction, and a set on less than instruction;

wherein the transforming and the lighting further include negating the graphics data and branching;

wherein a plurality of the operations are performed in parallel;

wherein the hardware graphics accelerator operates with an OpenGL application program interface.

54. (New) A method as recited in claim 45, wherein the operations further include a move, a multiply, an addition, a multiply and addition, a reciprocal, a reciprocal square root, a three component dot product, a four component dot product, a distance, a minimum, a maximum, a set on greater or equal than, an exponential, a logarithm, and a lighting.

55. (New) A method as recited in claim 45, wherein the graphics data includes vertex data, and the operations perform vertex processing on the vertex data.

56. (New) A method as recited in claim 55, wherein multiple vertices represented by the vertex data are operated upon in parallel.

57. (New) A method as recited in claim 45, wherein the graphics data is swizzled.

58. (New) A method as recited in claim 52, wherein the instruction set further includes a multiply and addition instruction, a reciprocal instruction, a reciprocal square root instruction, a three component dot product instruction, a four component dot product instruction, a distance instruction, a minimum instruction, a maximum instruction, a set on greater or equal than instruction, an exponential instruction, a logarithm instruction, and a lighting instruction.

59. (New) A method as recited in claim 52, wherein the graphics data includes vertex data, and the operations perform vertex processing on the vertex data.

60. (New) A method as recited in claim 59, wherein multiple vertices represented by the vertex data are operated upon in parallel.

61. (New) A method as recited in claim 52, wherein the graphics data is swizzled.

62. (New) A method as recited in claim 52, and further comprising:

determining whether the hardware graphics accelerator is operating in a programmable mode;

performing the operations on the graphics data if it is determined that the hardware graphics accelerator is operating in the programmable mode; and

operating on the graphics data in accordance with a standard graphics application program interface if it is determined that the hardware graphics accelerator is not operating in the programmable mode.